A New Simulated Annealing Algorithm for the Multiple Sequence Alignment Problem: The approach of Polymers in a Random Media

M. Hernández-Guía
*Henri-Poincaré Group of Complex Systems, Physics Faculty,
University of Havana, La Habana, CP 10400, Cuba and
National Bioinformatics Center, Industria y San José,
Habana Vieja, Capitolio Nacional, CP 10200, Cuba*

R. Mulet
*Henri-Poincaré Group of Complex Systems, Physics Faculty,
University of Havana, La Habana, CP 10400, Cuba and
Department of Theoretical Physics, Physics Faculty,
University of Havana, La Habana, CP 10400, Cuba*

S. Rodríguez-Pérez
*Henri-Poincaré Group of Complex Systems, Physics Faculty,
University of Havana, La Habana, CP 10400, Cuba and
Department of Informatics, University Center of Las Tunas* Vladimir Illich Lenin, *Las Tunas, CP 75200, Cuba*

(Dated: February 9, 2008)

We proposed a probabilistic algorithm to solve the Multiple Sequence Alignment problem. The algorithm is a Simulated Annealing (SA) that exploits the representation of the Multiple Alignment between D sequences as a directed polymer in D dimensions. Within this representation we can easily track the evolution in the configuration space of the alignment through local moves of low computational cost. At variance with other probabilistic algorithms proposed to solve this problem, our approach allows for the creation and deletion of gaps without extra computational cost. The algorithm was tested aligning proteins from the kinases family. When $D = 3$ the results are consistent with those obtained using a complete algorithm. For $D > 3$ where the complete algorithm fails, we show that our algorithm still converges to reasonable alignments. Moreover, we study the space of solutions obtained and show that depending on the number of sequences aligned the solutions are organized in different ways, suggesting a possible source of errors for progressive algorithms.

PACS numbers: 87.10.+e,87.15.cc,87.14.-g

I. INTRODUCTION

The Multiple Sequence Alignment (MSA) problem constitutes one of the fundamental research areas in Bioinformatics. While at first sight it may seem a simple extension of the two-string alignment problem *two strings good, four strings better*, for biologists, the multiple alignment of proteins or DNA is crucial in deducing their common properties [1]. Quoting Arthur Lensk [1, 2]: *One or two homologous sequences whisper... a full multiple alignment shouts out loud*.

In general, the sequences consist of a linear array of symbols from an alphabet of k-letters ($k = 4$ for DNA and $k = 20$ for proteins). Given D sequences to determine a good Multiple Sequence Alignment is a relative task. Usually one defines a score function that depends on the distances between the letters of the alphabet, and assumes that the better alignment is the one that minimizes this score function.

It is a common use to define the MSA score in terms of the scores of the pairwise global alignments of the sequences (Sum of Pairs score)[3]. Given two sequences $\vec{a} = a_1 \ldots a_m$ and $\vec{b} = a_1 \ldots b_m$ let $\Delta(a, b)$ be a cost of the mutation of a into b and γ the cost of inserting or deleting of the letter. Extending $\Delta(a, b)$ so that $\Delta(a, -) = \gamma$ and $\Delta(-, b) = \gamma$ and considering that a null (-) symbol isolated from others (-) pays an extra cost δ [3] we may define the score of a pairwise alignment $M_{i,j}$ for sequences a_i and b_j of size m as:

$$s(M_{i,j}) = \sum_{h=1}^{m} \Delta(a_{i,h}, b_{j,h}) + n\delta \tag{1}$$

where n is the number of isolated (-). Then, the score for the multiple alignment M is given by:

$$E(M_{i,j}) = \sum_{i,j} s(M_{i,j}) \tag{2}$$

The multiple sequence alignment has at least three important applications in Biology: classification of protein families and superfamilies, the identification and representation of conserved sequences features of both DNA or proteins that correlate structure and/or function and the deduction of the evolutionary history of the sequences studied [1, 4].

Unfortunately the problem is known to be *NP-complete* and no complete algorithm exist to solve real or random instances. Therefore, many heuristic algorithms have been proposed to solve this problem. The algorithm of Carrillo-Lipman [5] (which is complete), is a Dynamic Programming algorithm able to find the multiple alignment of 3 sequences, and with some heuristic added, to find the alignments, in reasonable time, of up to 6 sequences [1]. However, its computational cost scales very fast with the number of sequences and is of little utility for more ambitious tasks. In the first 90's the problem was approached using ideas coming from physics, J. Kim and collaborators [6] and M. Ishikawa and collaborators [7] used different version of the Simulated Annealing [8] technique with some success, but their algorithms were unable to change the number of gaps in the alignment. This means that once they started with a given initial configuration (usually taken from some heuristics), any motion of segments in the sequences conserved the number of gaps. To extend these programs allowing the number of gaps to change will cause the appearance of global moves in the algorithm that are very expensive from the computational point of view.

Probably the must successful attempt to solve this problem has been the Clustal project [9], a progressive algorithm that first organizes the sequences according to their distances and then aligns the sequences in a progressive way, starting with the most related ones. Moreover, it uses a lot of biological information, some motifs of residues rarely accept gaps, sub-sequences of residues associated with structural sub-units are preferred to stay together during the alignment, etc. These features, and a platform easy to use and integrated with other standard bioinformatic tools, have made Clustal the favorite Multiple Sequence Alignment program for biologists and people doing bionformatics in general [10]. However, it also has important drawbacks. Once the first k sequences are aligned, the inclusion of a new sequence would not change the previous alignment, the gap penalties are the same independently on how many sequences have been already aligned or their properties, and being a progressive method the global minimum obtained is strongly biased by those sequences which are more similar[9].

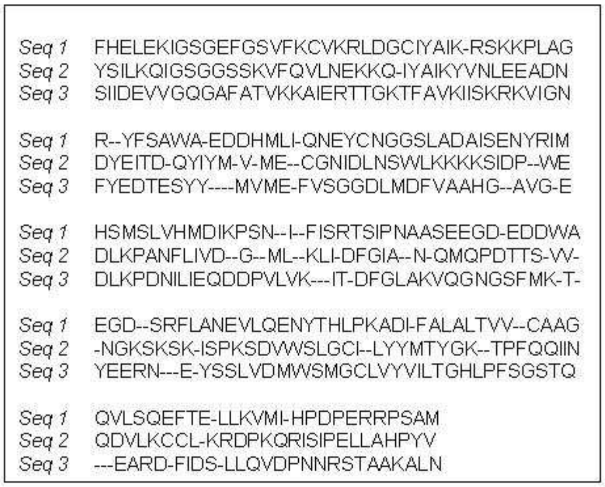
Another, recent and also successful approach uses the concepts of Hidden Markov Models [11]. While some of the previous drawback associated to Clustal disappear, because for example, the sequences do not need to be organized a priori, one most start assuming a known model of protein (or DNA) organization, which is usually obtained after training the program in a subset of sequences. Then, one must be aware that the results usually depend on the training set, specially if it is not too large. Moreover, if we are dealing with sequences of unknown family, or difficult to be characterized this approach does not guarantee good alignments.

Therefore we decided to propose a new Simulated Annealing (SA) algorithm that avoids the main difficulty of the previous attempts [6, 7]. Our algorithm allows for the insertion and deletion of gaps in a dynamic way using only local moves. It makes use of the mathematical mapping between the Multiple Sequence Alignment and a Directed Polymer in a Random Media (DPRM) pointed out some years ago by Hwa et al [12]. In such a way, it should be also possible to extrapolate all the computer facilities and techniques, developed in the field of polymers to this biological problem.

The rest of the paper is organized in the following way. In the next section we make a short review of the theoretical foundations of our algorithm. Then in section III we explain the implementation details to discuss the results in section IV. Finally the conclusions are presented including and outlook for future improvements of this program.

II. THEORETICAL BACKGROUND

Usually, multiple sequence alignments are studied and visualized writing one sequence on the top of the other, miming a table, (see figure 1) and all the probabilistic algorithms devised so far use the simplicity of this representation to generate the moves.



```
Seq 1   FHELEKIGSGEFGSVFKCVKRLDGCIYAIK-RSKKPLAG
Seq 2   YSILKQIGSGGSSKVFQVLNEKKQ-IYAIKYVNLEEADN
Seq 3   SIIDEVVGQGAFATVKKAIERTTGKTFAVKIISKRKVIGN

Seq 1   R--YFSAWA-EDDHMLI-QNEYCNGGSLADAISENYRIM
Seq 2   DYEITD-QYIYM-V-ME--CGNIDLNSWLKKKKSIDP--WE
Seq 3   FYEDTESYY----MVME-FVSGGDLMDFVAAHG--AVG-E

Seq 1   HSMSLVHMDIKPSN--I--FISRTSIPNAASEEGD-EDDWA
Seq 2   DLKPANFLIVD--G--ML--KLI-DFGIA--N-QMQPDTTS-VV-
Seq 3   DLKPDNILIEQDDPVLVK---IT-DFGLAKVQGNGSFMK-T-

Seq 1   EGD--SRFLANEVLQENYTHLPKADI-FALALTVV--CAAG
Seq 2   -NGKSKSK-ISPKSDVWSLGCI--LYYMTYGK--TPFQQIIN
Seq 3   YEERN---E-YSSLVDMWSMGCLVYVILTGHLPFSGSTQ

Seq 1   QVLSQEFTE-LLKVMI-HPDPERRPSAM
Seq 2   QDVLKCCL-KRDPKQRISIPELLAHPYV
Seq 3   ---EARD-FIDS-LLQVDPNNRSTAAKALN
```

FIG. 1: Usual representation of a multiple sequence alignment

Instead of that, we will use the well known fact [3, 16] that the alignment of D sequences may be represented in a D dimensional lattice (see figure 2 for $D = 2$).

The cells of the D-dimensional lattice are labeled by the D indexes $(i_1, i_1 \dots i_D)$. The bonds encode the adjacency of letters: A diagonal bond in a D dimensional space represents the D-pairing $(a_{i_1}, b_{i_2}, \dots, W_{i_D})$. The insertion of gaps are represented by bonds without components in the sequences where the gaps were inserted. For example, a D-pairing $(a_{i_1}, b_{i_2}, -, d_{i_4}, \dots, W_{i_D})$ is represented by a bond whose projection on the third sequence is zero, and the D-pairing $(a_{i_1}, -, -, d_{i_4}, \dots, W_{i_D})$ is represented by a bond whose projection on the second and third sequences are zero.

Then, any alignment maps onto a lattice path that is directed along the diagonal of the D-dimensional hypercube. This lattice path may be interpreted as a Directed Polymer and the Random Media in the problem is provided by the structure of the sequences to be aligned and by the distance between the residues in the different sequences.

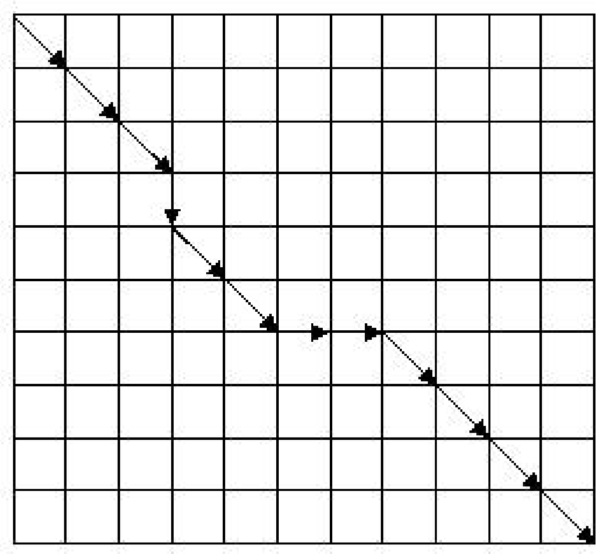


FIG. 2: A directed path (thick line) in a bi-dimensional grid

This mapping was already fruitfully used by Hwa and co.[12] to prove that the similarities between two sequences can be detected only if their amount exceeds a threshold value and for proposing a dynamic way to determine the optimal parameters for a good alignment of two sequences.

Here, our main focus will be to optimize the Directed Polymer (lattice path) under the constraints imposed by the sequences and their interactions in dimensions larger than 2. To use a Simulated Annealing algorithm we extend the usual representation of computer science of determining the ground state of the problem to a finite temperature description. Then, a finite-temperature alignment is a probability distribution

$$P(C) = \frac{1}{Z} e^{-\beta E(C)} \tag{3}$$

over all possible conformations C of the polymer and where E is given by equation 2, and Z is the partition function of the alignment [13]. The temperature (β^{-1}) controls the relative weight of alignments with different scores (different conformations of the polymer) while $\Delta(a,b)$ the length of the polymer and the frequency of the gaps. In physical terms, $P(C)$ defines and ensemble at temperature β^{-1} with line tension γ and chemical potential $\Delta(a,b)$.[14]

III. THE ALGORITHM

The Simulated Annealing (SA) was introduced many years ago by Kirkpatrick et al [8] to find a global minimum of a function in combinatorial optimization problems. SA is a probabilistic approach, that in general needs a state space (the different configurations of the directed path) and a cost or energy function (2) to be minimize (eq. 2).

Simulated Annealing generates new alignments from a current alignment by applying transition rules of acceptance. The criteria for acceptance are the following:

- if $\Delta E < 0$, accept the new alignment
- if $\Delta E > 0$ accept it with probability $P(\Delta E) = e^{-\beta(E_{new}(C) - E_{old}(C))}$

The parameter β controls the probability to accept a new configuration. Initially, one starts at low values of β (high temperatures) and then increases it applying an annealing schedule. If the temperature is lowered slowly enough, it can be proved that the system reaches a global minimum [15]. Unfortunately it will require infinitely computational time and one usually selects the best scheduling that is possible to afford with the computational facilities at hand. Then SA is run over many initial conditions, and one assumes that the output of minimum energy is (or is close) to the global minimum.

In the Multiple Sequence Alignment this is also the case, but differently to what happens in other combinatorial optimization problems, here the average solution, i.e. that obtained after averaging over all the local minima's may be interesting by itself. In fact, researchers are often interested not in the particular details of the alignment, but in its robust properties, and comparing all the outputs of the SA is a way to get this information.

From the technical point of view, once a cost function is defined, one needs to select the moves to be associated to the transition rates. Our description of the Multiple Sequence Alignment Problem as a Directed Polymer in a Random Media allows us to define three types of moves, insertion, deletion and motion of gaps. All these moves are represented in figure (3) in a two-dimensional grid. The extension to D-dimensional systems is straightforward.

In this way we get an algorithm that allows for the creation of gaps, which means a search space larger than the usually proved by similar methods. At the same time the algorithm is quadratic in the number of sequences. In fact, the computational cost of any move is limited by the square of the number of sequences to be aligned.

In this work, we did not follow any heuristic strategy of optimization. Our intention was to prove the potentiality of this strategy and we kept things as simple as possible. For example, if we start too far from the global minimum, the selection of local moves alone will make the algorithm to converge very slowly to it. This drawback may be overcome using very different initial conditions or

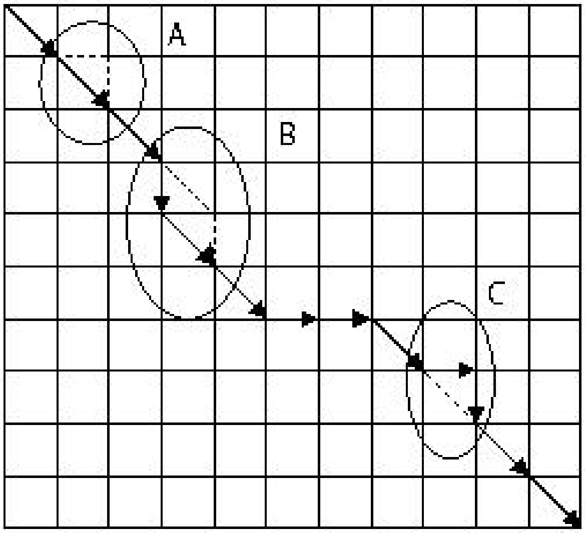


FIG. 3: Local moves of the algorithm in a two dimensional grid (from arrows to dashed lines), a) gap insertion, b) gap motion and c) gap deletion

trying, every some time steps, global moves that change radically the conformation of the polymer. We did not take care of this. During the simulation the three moves were chosen with probability 1/3. The only biological information inserted was given by the cost matrix used to align the protein sequences. We avoid the use of important and well know biological information, fixed residues, phylogenetic tree of the sequences, etc, and the program was designed without the use programming optimization tricks.

IV. RESULTS

All the results presented in this section reflects the alignments of proteins from the kinase family, but qualitatively similar results were obtained for the GPCRs (G protein-coupled receptors) and CRP (cAMP receptor protein) families. The simulations started with $\beta = 1.0$ and every t_{rel} montecarlo steps β was increased by a multiplicative factor of 1.01 until $\beta = 3.0$. We took care and in all cases the system reached the equilibrium. The different initial conditions were chosen inserting gaps randomly in all the sequences but the larger one, such that considering these gaps at $t = 0$, all the sequences were of equal length. To define the distance between the letters of the alphabet we use the PAM matrix [11].

In figure 4 are shown the approach to equilibrium of the multiple sequence alignment of 3 sequences averaged over 100 initial conditions for $t_{rel} = 10, 100, 1000$ and 10000. Comparing with the result of the Carrillo-Lipman algorithm it is evident that for $t_{rel} = 10000$ the algorithm is very close to the global minimum. However, one should take care that differently to what is usually obtained in other algorithms for the Multiple Sequence Alignment problem, these figures reflects averages values of the multiple sequence alignment.

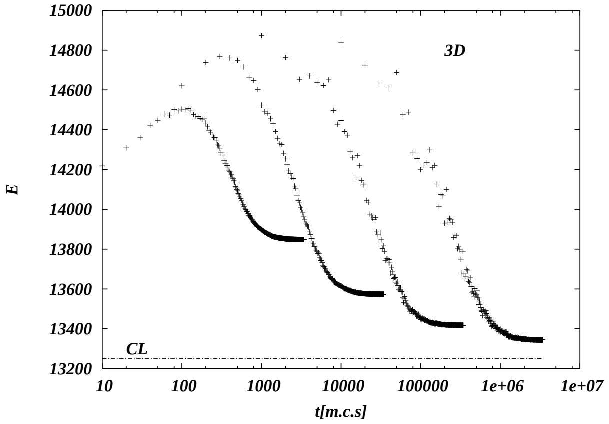


FIG. 4: Mean Energy versus time for the alignment of three sequences from the kinase family. From left to right: $t_{rel} = 10, 100, 1000$ and 10000. The average were taken over 100 initial conditions. The horizontal line represents the result of the Carrillo-Lipman algorithm

In fact, one is often interested, rather than on the average, in the minimum of all the alignments. In figure 5 we represent an histogram in energy for the alignment, over 1000 initial conditions for different values of t_{rel}, of the same three proteins of the kinase family presented in figure 4. Note again that for $t_{rel} = 1000$ we obtain exactly the Carillo-Lipman result with probability larger than 0. Moreover, looking to the structure of the histogram for $t_{rel} = 1000$, one can also conjecture that if the average multiple sequence alignment were calculated only with those alignments concentrated in the peak of lower energies, the result presented in figure 4 will be closer to the one of the Carrillo-Lipman algorithm.

Another symptom suggesting that the average over the realization must be taken with care comes from the analysis of figure 6. There we present again histograms for $t_{rel} = 1000$ but using three different samples of the kinase family. Note that while sample 1 and sample 3 are very well behaved and the results compare very well with the Carillo-Lipman method, the situation for sample 2 is different. To get good results in this case, it is clearly necessary to go beyond 1000 montecarlo steps.

In the same spirit of figure 4, figures 7 and 8 reflect results suggesting that also for higher dimensions, if t_{rel} is large enough the algorithm should produce good alignments. In fact, also in these cases the energy decreases linearly with t_{rel}.

If the number of sequences is higher, the correlations between the sequences increases, and the algorithm should find better results. This fact may be clearly seen in figure 9 where the equilibration time of the algorithm

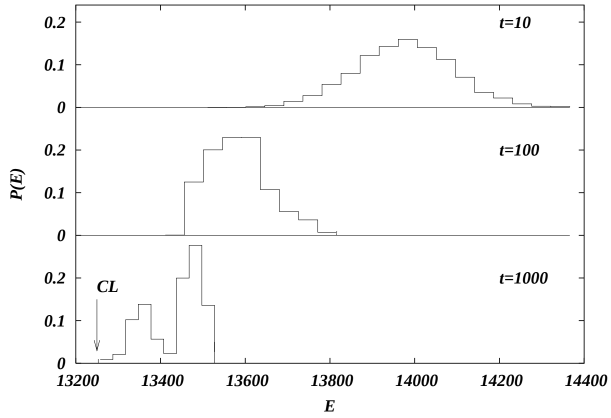


FIG. 5: Histograms of energy for the alignment of three sequences from the kinase family at different t_{rel}.

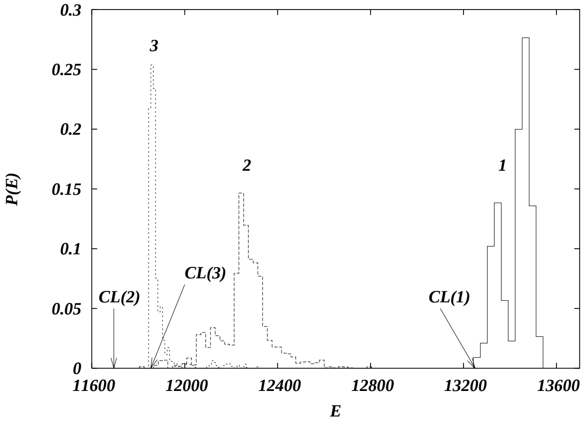


FIG. 6: Histograms of energy for the alignment of three different samples of three sequences from the kinase family at $t_{rel} = 1000$. The Carrillo-Lipman results are represented by the arrows.

(in m.c.s) is shown as a function of the number of sequences. The equilibration time, measured as the time necessary to reduce the energy by a factor e, decreases linearly with the number of sequences to be aligned. Of course the results may change if very different sequences are aligned, but in this case all other known algorithms fail to predict good alignments. Then, we may say that for the most common cases, of correlated sequences, we present an algorithm whose convergence time decreases with D, and whose moves, only increase quadratically with D.

With these results at hand, we follow to study the structure of the space of the solutions as a function of the number of aligned sequences. We define a distance (d) between two alignments $\mathcal{A}$, and $\mathcal{B}$ in the following way. Given two solutions, $A_{i,j}$ and $B_{i,j}$ (where the index i stands for the sequence and j for the position of the symbol in the sequence) we, aligned one by one of the D sequences of each solution using a Dynamic Programming algorithm reminiscent of the Needleman-Wunsh algorithm with the following score function:

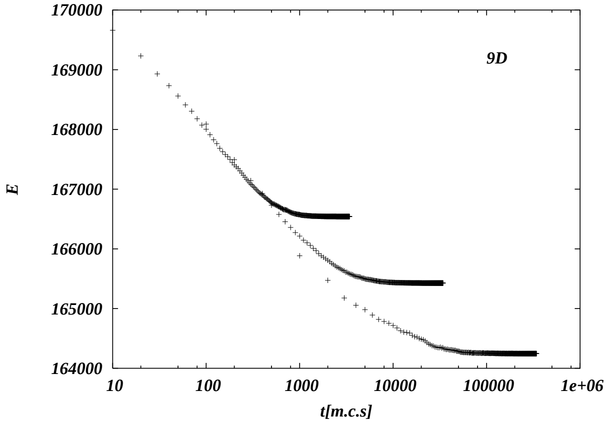


FIG. 7: Mean Energy versus time for the alignment of nine sequences from the kinase family. From left to right: $t_{rel} = 10, 100$ and 1000. The average were taken over 100 initial conditions.

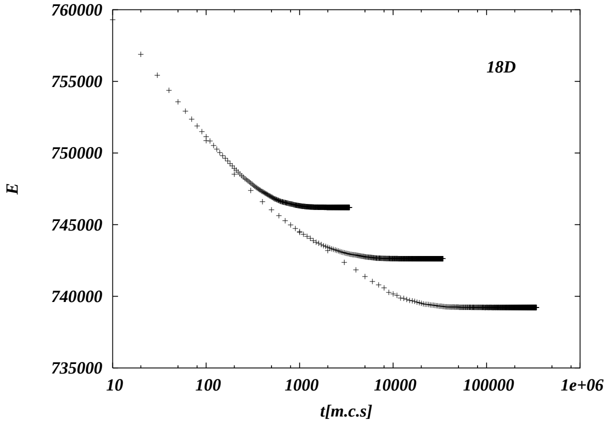


FIG. 8: Mean Energy versus time for the alignment of 18 sequences from the kinase family. From left to right: $t_{rel} = 10, 100$ and 1000. The average were taken over 100 initial conditions.

$$c_a(A, B) = \begin{cases} 0 & \text{if } A_{i,j} = B_{i,j} \\ 1 & \text{if } A_{i,j} \neq B_{i,j} \\ r > 1 & \text{if a gap is inserted} \end{cases}$$

and express $d_{\mathcal{A},\mathcal{B}}$ as:

$$d_{\mathcal{A},\mathcal{B}} = \frac{1}{D} \sum_{i,j} c_a(A_{i,j}, B_{i,j}) \qquad (4)$$

In this way identical alignments will be at distance 0 from each other, and the insertion of gaps to obtain good

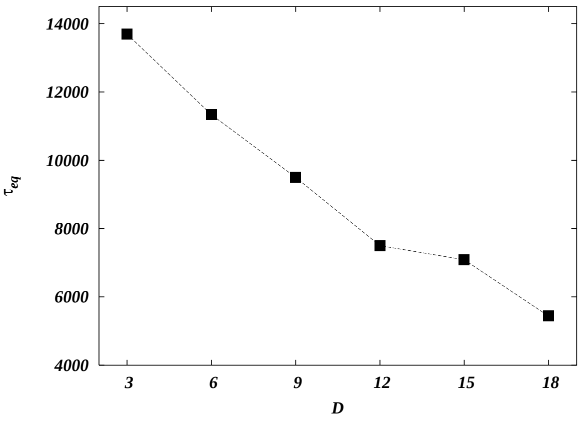


FIG. 9: Equilibration time versus D average over 500 initial conditions for $t_{rel} = 1000$.

alignments is penalized, such that the original alignments are altered minimally during the calculation of $d_{\mathcal{A},\mathcal{B}}$. We calculated d with $r = 2$ and $r = 8$ and the results were the same (apart a constant shift in d). Below we present figures for $r = 2$.

We study how different are, from the solution of minimum score, the other solutions obtained aligning D sequences. For every value of D we use 1000 initial conditions and a $t_{rel} = 1000$. Note that the sequences used were always the same. This mean, we first aligned three sequences from the kinase family. Then, to align 4 sequences we just add a new one to the previous three and started the alignment from scratch. The procedure was repeated for every new value of D.

The results appear in figures 10 and 11 where $E - E_{min}$ is plotted as a function of d. From the figures it becomes evident that the space of solution strongly depends on the number of sequences aligned. For example, while for three sequences the distance between the alignments is correlated with the difference in score, for more than 4 sequences it is not true anymore. Moreover, while for $D < 6$ the distance between the solutions decreases, it increases for $D > 6$ and remains constant for $D > 12$. Surely, these results reflect the correlation between the proteins aligned. For example, we may speculate that for $3 < D < 6$, any new protein added contributed to find more similar alignments, i.e. added relevant information to the system. However, for $D > 6$ the sequences contributed with new and uncorrelated information that produce more distant alignments and for $D > 12$ to add new proteins that belong to the kinase family will not change the relevant characteristics of the alignment.

These results are relevant, considering the limitations of progressive algorithms to change the previous alignment when new sequences are added. Figures 10 and 11, clearly suggest that the inclusion of one single sequence may dramatically change the character of the solutions.

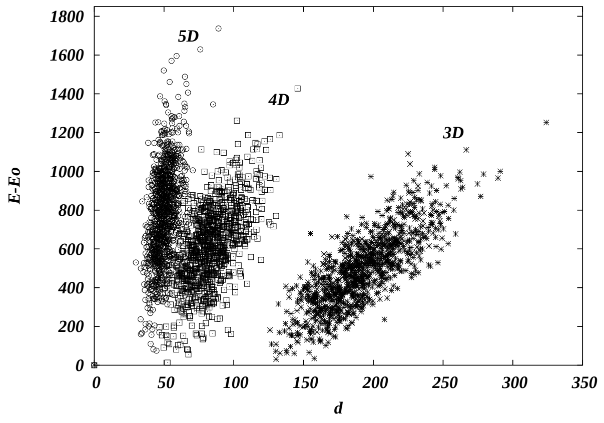


FIG. 10: $E - E_0$ versus d for the alignment of 3,4 and 5 sequences of the kinase family

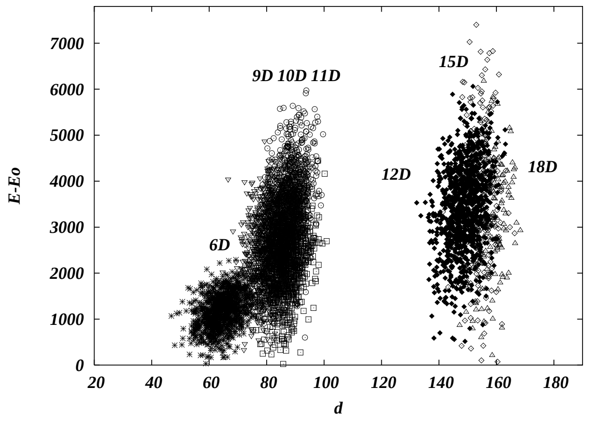


FIG. 11: $E - E_0$ versus d for the alignment of 6,9,10,11,12,15 and 18 sequences of the kinase family

V. CONCLUSIONS AND OUTLOOK

In this work we presented a new probabilistic algorithm, to perform the Multiple Alignment of proteins. The algorithm is based on the mapping between the DPRM and the Multiple Sequence Alignment problem. At variance with other probabilistic algorithms our algorithm permits the variation of the number of gaps in the alignment without the necessity of expensive global moves. It is proved that for small number of sequences it reproduces the results of a complete algorithm. Moreover, we show that for practical purposes the equilibration time is almost independent on the number of sequences aligned D and in the worst case, it scale linearly with D. Finally, we study the space of solutions for different number of aligned sequences, and find a very rich structure that indicates the importance of just one sequence in the multiple alignment.

Of course the algorithm is still far from being competitive with other approaches like HMM and CLUSTAL. We are already working in implementing a similar work but using Parallel Tempering instead of Simulated Annealing. It is know that Parallel Tempering is more useful that S.A. when dealing with very hard problems, like spin glasses. Moreover, it is very suitable to parallelization. Also a direction of current work is the introduction of biological information relevant for the alignment. This may impose important constraints in the possible alignments, that may in turn strongly reduce the space of possible solutions. And last, but not least, important programing optimization are necessary to make competitive this program from the computing time point of view.

Acknowledgments

R. M. will like to thank K. Leon and E. Moreno for useful discussion at the early stage of this work.

[1] D. Gunsfield, *Algorithms on Strings, Trees and Sequences* (1999) Cambridge University Press.

[2] T. J. P. Hubbard, A. M. Lensk and A. Tramontano, Nature Structural Biology **4**, 313 (1996)

[3] M.S. Waterman, *Introduction to Computational Biology* (200), Chapm & Hall.

[4] P. Pevzner, *Computational Molecular Biology* (2000) MIT Press.

[5] H. Carrillo and D. Lipman, SIAM J. Appl. Math, **48**, 1073, (1988)

[6] J. Kim, S. Pramanik and M. J. Chung, CABIOS **10** 419 (1994)

[7] M. Ishikawa, T. Toya, M. Hoshida, K. Nitta, A. Ogiwara and M. Kanehisa, CABIOS **9** 267, (1993)

[8] S. Kirkpatrick, C.D. Gelatt and M. P. Vecchi, Science **220**, 671 (1983)

[9] J. D. Thompson, D. G. Higgings and T. J. Gibson, Nucleid Acids Res. **22**, 4673 (1994)

[10] We are not aware of any statistical study proving this, but we believe that this statment is well accepted within the colleagues we contacted.

[11] R. Durbin, S. R. Eddy, A. Krogh and G. Mitchison, *Biological Sequence Analysis* (1998) Cambridge University Press.

[12] T. Hwa and Michael Lassig, Phys. Rev. Lett. **76**, 2591 (1996)

[13] M. Q. Zhang and T. G. Marr, J. Theor. Biol. **174**, 119 (1995)

[14] M. Kschischo and M. Lässig, preprint

[15] S. Geman, S. Geman, IEEE transictions on Pattern Analysis and Machine Inteligence, **6**,721 (1984)

[16] S.B. Needleman and C. D. Wunsch, J. Mol. Biol. **48**, 444 (1981)